Fulfillment API Overview

You can use the Fulfillment APIs to process shipments through their fulfillment processes by placing a series of calls identifying the action you want to perform on the shipment.

This guide provides a reference for shipment data such as the supported shipment methods and service types, as well as explains the useful information returned in the Shipment API responses that allows the user to identify the next actions available for the shipment. Examples of STH, BOPIS, and Transfer API fulfillment processes are included. See the full API documentation for the comprehensive schemas of Shipment API and Fulfillment calls.

For more information about fulfilling shipments in general, see the Fulfillment Methods guide.

Shipping Methods and Service Types

Shipping method codes include a prefix that indicates the carrier followed by the service type. If a customer was not allowed to choose a specific carrier or a shipping method, or if the proper codes are unknown by the user, a generic Kibo code can be sent instead. In that case, the system will map the generic value to the appropriate code. These generic codes are:

- Kibo Generic: KIBO_STANDARD, KIBO_1_DAY, KIBO_2_DAY, and KIBO_3_DAY
- USPS: KIBO_USPS_STANDARD, KIBO_USPS_1_DAY, KIBO_USPS_2_DAY, and KIBO_USPS_3_DAY
- UPS: KIBO_UPS_STANDARD, KIBO_UPS_1_DAY, KIBO_UPS_2_DAY, and KIBO_UPS_3_DAY
- FedEx: KIBO_FEDEX_STANDARD, KIBO_FEDEX_1_DAY, KIBO_FEDEX_2_DAY, and KIBO_FEDEX_3_DAY
- Canada Post: KIBO_CANADAPOST_STANDARD, KIBO_CANADAPOST_1_DAY, KIBO_CANADAPOST_2_DAY, and KIBO_CANADAPOST_3_DAY
- Purolator: KIBO_PUROLATOR_STANDARD, KIBO_PUROLATOR_1_DAY, KIBO_PUROLATOR_2_DAY, and KIBO_PUROLATOR_3_DAY

Expand the accordions below to view the full list of supported service types for each carrier.

USPS Service Types

- STANDARD POST
- GROUND
- LIBRARY
- FIRST_CLASS INTERNATIONAL
- PRIORITY INTERNATIONAL
- EXPRESS INTERNATIONAL
- MEDIA
- EXPRESS

- FIRST_CLASS
- PRIORITY

UPS Service Types

- UPS_GROUND
- UPS_STANDARD
- UPS_SUREPOST_BPM
- UPS_SUREPOST_MEDIA
- UPS_SUREPOST_1LB_OR_GREATER
- UPS_SUREPOST_LESS_THAN_1LB
- UPS_WORLDWIDE_EXPEDITED
- UPS_WORLDWIDE_EXPRESS_PLUS
- UPS_WORLDWIDE_EXPRESS_FREIGHT
- UPS_EXPRESS
- UPS_NEXT_DAY_AIR
- UPS_NEXT_DAY_AIR_EARLY
- UPS_NEXT_DAY_AIR_SAVER
- UPS_SECOND_DAY_AIR_AM
- UPS_SECOND_DAY_AIR
- UPS_SAVER
- UPS_THREE_DAY_SELECT

FedEx Service Types

- FEDEX_1_DAY_FREIGHT
- FEDEX_2_DAY
- FEDEX_2_DAY_AM
- FEDEX_2_DAY_FREIGHT
- FEDEX_3_DAY_FREIGHT
- FEDEX_DISTANCE_DEFERRED
- FEDEX_EXPRESS_SAVER
- FEDEX_FIRST_FREIGHT
- FEDEX_FREIGHT_ECONOMY
- FEDEX_FREIGHT_PRIORITY

- FEDEX_GROUND
- FEDEX_NEXT_DAY_AFTERNOON
- FEDEX_NEXT_DAY_EARLY_MORNING
- FEDEX_NEXT_DAY_END_OF_DAY
- FEDEX_NEXT_DAY_FREIGHT
- FEDEX_NEXT_DAY_MID_MORNING
- FIRST_OVERNIGHT
- GROUND_HOME_DELIVERY
- INTERNATIONAL_ECONOMY
- INTERNATIONAL_ECONOMY_FREIGHT
- INTERNATIONAL_FIRST
- INTERNATIONAL_PRIORITY
- INTERNATIONAL_PRIORITY_FREIGHT
- PRIORITY_OVERNIGHT
- SAME_DAY
- SAME_DAY_CITY
- SMART_POST
- STANDARD_OVERNIGHT

FedEx Cross Border Service Types

- fedexcrossborder_FDXIE (International Express)
- fedexcrossborder_FDXIP (International Priority)

Canada Post Service Types

- canadapost_Xpress_Post
- canadapost_Expedited_Parcel

Purolator Service Types

- purolator_PurolatorGround
- purolator_PurolatorQuickShip
- purolator_PurolatorExpress

Querying Order Shipments

Before getting started, know that shipments get created automatically when an order is accepted. Upon submission, the order will automatically transition through the stages of Submit > Validate > Accept.

You can retrieve the shipments for a particular order by querying shipments for `orderId`. the Shipment API has different sorting and filtering syntax than most other APIs:

```
GET api/commerce/shipments?filter=orderId=={orderID}
```

You can include other filters if you don't care about certain statuses or shipment types:

```
GET api/commerce/shipments?filter=orderId==
{orderId};shipmentStatus!=REASSIGNED;shipmentStatus!=CANCELED;shipmentType!=Transfer
```

Note: STH Consolidation

For cases that use STH Consolidation, there are some additional fields to be aware of when making calls related to fulfillment.

- The parent shipment being consolidated will have a `consolidatedLocationRoute` field set as the shopper's delivery location. The `shipTo` field is also the shopper's delivery location while `shipFrom` is the centralized location fulfilling the shipment.
- The transfer child shipments will have `shipTo` as the parent consolidation location and `shipFrom` as the transfer location, as well as a `transferConsolidated` flag.
- The Fulfillment API documentation has not yet been updated to include these fields (as of June 2022), but they will be added soon.

Additionally, whether or not a location is enabled for STH Consolidation is linked to the `shipToHomeConsolidation` flag in the Location APIs.

Shipment Response Actions

When using the Shipment API to transition a shipment through each stage of its fulfillment process, it can be difficult to remember how to format the next step's endpoint and the expected parameters on-the-fly. Additionally, if a call such as cancellation, adding tracking information, or marking the shipment as fulfilled needs to be performed outside of the usual fulfillment flow, it may be confusing to determine how to perform the action without referring to the documentation. To assist with this, the Shipment API provides guidelines within the response body.

When making a call to the Shipment API, including any of the calls detailed in the STH, BOPIS, or Transfer fulfillment flows, the response is usually the full record of shipment data. This response includes two objects, *workflowState* and *_links*, that are very useful to be aware of when fulfilling shipments via API. These data elements, which usually appear at the end of the response, are shown below as an example returned for a STH shipment that has just been created and

not yet accepted by the assigned location. The tenantID, host, and shipment number variables would all be populated with the appropriate information for the user and shipment by the API.


```
"workflowState": {
      "shipmentState": "PRE_ACCEPT_SHIPMENT",
      "taskList": [
         {
            "taskId": "2162",
            "name": "Accept Shipment",
            "subject": "",
            "description": "Accept shipment",
            "skippable": false,
            "inputs": [
               {
                  "name": "shipmentAccepted",
                  "required": true,
                  "label": "Accept Shipment?",
                  "helpMessage": "",
                  "type": "BOOLEAN"
               }
            ],
            "active": true,
            "completed": false,
            "_links": {
               "execute": {
                  "href": "https://t./api/commerce/shipments/{shipmentNumber}/tasks/Accept%20Sh
ipment/completed",
                  "method": "PUT"
               }
            }
         },
         {
            "name": "Validate Items In Stock",
            "subject": "",
            "inputs": [
               {
                  "name": "stockLevel",
                  "type": "STRING"
               }
            ],
            "active": false,
            "completed": false,
            "_links": {}
         },
         {
            "name": "Print Packing Slip",
            "subject": "",
            "inputs": [
               {
                  "name": "back",
                  "type": "BOOLEAN"
               }
            ],
            "active": false,
            "completed": false,
            "_links": {}
         },
         {
            "name": "Prepare for Shipment",
```

```
        "subject": "",
        "inputs": [
          {
            "name": "canceled",
            "type": "BOOLEAN"
          },
          {
            "name": "back",
            "type": "BOOLEAN"
          }
        ],
        "active": false,
        "completed": false,
        "_links": {}
      }
    ],
    "processInstanceId": "1251",
    "auditInfo": {
      "updateDate": "2020-03-16T18:26:21.562Z",
      "createDate": "2020-03-16T18:26:21.022Z",
      "updateBy": "",
      "createBy": ""
    }
  },
  "changeMessages": [],
  "data": {
    "atgOrderId": "o1923076176",
    "ordershippingtaxtotal": "0",
    "OrderType": "STH",
    "ordershippingtotal": "7.95",
    "customerNumber": "9990042837414",
    "originalTenderType": "VI",
    "returnsBarcode": "00700001007674401021820"
  },
  "email": "example@email.com",
  "isExpress": false,
  "auditInfo": {
    "updateDate": "2020-03-16T18:26:21.562Z",
    "createDate": "2020-03-16T18:26:21.022Z",
    "updateBy": "",
    "createBy": ""
  },
  "_links": {
    "self": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}"
    },
    "shipments": {
      "href": "https://t./api/commerce/shipments"
    },
    "tasks": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}/tasks"
    },
    "workflowInstanceImage": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}/workflow-instance-image"
    },
    "backorder": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}/backordered",
      "method": "POST"
    },
    "backorderItems": {
```

```
    "backorderItems": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}/backorderedItems",
      "method": "POST"
    },
    "reject": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}/rejected",
      "method": "PUT"
    },
    "rejectItems": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}/rejectedItems",
      "method": "PUT"
    },
    "cancel": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}/canceled",
      "method": "PUT"
    },
    "fulfill": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}/fulfilled",
      "method": "PUT"
    },
    "reassign": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}/reassigned",
      "method": "PUT"
    },
    "reassignItems": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}/reassignedItems",
      "method": "PUT"
    },
    "cancelItems": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}/canceledItems",
      "method": "PUT"
    }
```

Workflow State

The workflowState object not only provides information about the current state that the shipment is in, but also lists the upcoming “tasks” of the fulfillment flow that the shipment is expected to follow. These tasks are the steps in the fulfillment flow that the shipment must move through in order to be completed. For this STH example, the tasks are Accept Shipment, Validate Items in Stock, Print Packing Slip, and Prepare for Shipment. Providing a preview of this flow helps the user know which actions to take to transition the shipment through fulfillment.

Each task at least has a*name*, *inputs* (or, the template for the request body), and booleans indicating whether that task is *active* (whether the shipment is currently in that state) and *completed* (whether the shipment has already passed through that state). The task that is active displays more detailed information, such as an ID, whether the task can be skipped according to the fulfillment flow, and the execution link of the task. This task-level *_links* object provides the endpoint and HTTP method of the call that will be made to complete this step and move the shipment to the next state. Combined with the *inputs* that define the request body, the user has all of the information that they need to make the call and continue with fulfillment.

```
"workflowState": {
    "shipmentState": "PRE_ACCEPT_SHIPMENT",
    "taskList": [
      {
        "taskId": "2162",
        "name": "Accept Shipment",
        "subject": "",
```

```
            "description": "Accept shipment",
            "skippable": false,
            "inputs": [
               {
                  "name": "shipmentAccepted",
                  "required": true,
                  "label": "Accept Shipment?",
                  "helpMessage": "",
                  "type": "BOOLEAN"
               }
            ],
            "active": true,
            "completed": false,
            "_links": {
               "execute": {
                  "href": "https://t./api/commerce/shipments/{shipmentNumber}/tasks/Accept%20Sh
ipment/completed",
                  "method": "PUT"
               }
            }
         },
         {
            "name": "Validate Items In Stock",
            "subject": "",
            "inputs": [
               {
                  "name": "stockLevel",
                  "type": "STRING"
               }
            ],
            "active": false,
            "completed": false,
            "_links": {}
         },
         {
            "name": "Print Packing Slip",
            "subject": "",
            "inputs": [
               {
                  "name": "back",
                  "type": "BOOLEAN"
               }
            ],
            "active": false,
            "completed": false,
            "_links": {}
         },
         {
            "name": "Prepare for Shipment",
            "subject": "",
            "inputs": [
               {
                  "name": "canceled",
                  "type": "BOOLEAN"
               },
               {
                  "name": "back",
                  "type": "BOOLEAN"
               }
```

```
            ],
            "active": false,
            "completed": false,
            "_links": {}
        }
    ],
    "processInstanceId": "1251",
    "auditInfo": {
        "updateDate": "2020-03-16T18:26:21.562Z",
        "createDate": "2020-03-16T18:26:21.022Z",
        "updateBy": "",
        "createBy": ""
    }
},
"changeMessages": [],
"data": {
    "atgOrderId": "o1923076176",
    "ordershippingtaxtotal": "0",
    "OrderType": "STH",
    "ordershippingtotal": "7.95",
    "customerNumber": "9990042837414",
    "originalTenderType": "VI",
    "returnsBarcode": "00700001007674401021820"
},
"email": "example@email.com",
"isExpress": false,
"auditInfo": {
    "updateDate": "2020-03-16T18:26:21.562Z",
    "createDate": "2020-03-16T18:26:21.022Z",
    "updateBy": "",
    "createBy": ""
},
```


This workflow information is returned by any GET Shipment API call, as well as included in the response when editing a shipment or making any of the calls to transition it through the fulfillment states. The workflow information updates with each new response after tasks are performed and the shipment changes states. So for this example case after making the Accept Shipment call, the response would list the Accept Shipment task as *active = false* and *completed = true*, and the Validate Items in Stock task would be the new active task with its *_links* information populated.

Links

While the task-level *_links* object shown above in the *workflowState* element only lists the call information for that particular task, the shipment-level *_links* object contains a list of all other possible actions that the user may want to perform on the shipment. Like *workflowState*, this information is returned by any GET Shipment API call as well as included in the response when editing a shipment or making any of the calls to transition it through the fulfillment states.

```
  "_links": {
    "self": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}"
    },
    "shipments": {
      "href": "https://t./api/commerce/shipments"
    },
    "tasks": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}/tasks"
    },
    "workflowInstanceImage": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}/workflow-instance-image"
    },
    "backorder": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}/backordered",
      "method": "POST"
    },
    "backorderItems": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}/backorderedItems",
      "method": "POST"
    },
    "reject": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}/rejected",
      "method": "PUT"
    },
    "rejectItems": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}/rejectedItems",
      "method": "PUT"
    },
    "cancel": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}/canceled",
      "method": "PUT"
    },
    "fulfill": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}/fulfilled",
      "method": "PUT"
    },
    "reassign": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}/reassigned",
      "method": "PUT"
    },
    "reassignItems": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}/reassignedItems",
      "method": "PUT"
    },
    "cancelItems": {
      "href": "https://t./api/commerce/shipments/{shipmentNumber}/canceledItems",
      "method": "PUT"
    }
```


This list is static and does not change, regardless of what state the shipment is currently in, so it is useful as a reference for events such as cancellation that do not require the shipment to be in a particular step. It can also be used for cases that do not follow the best-case scenario projected in the basic workflow, such as backordering or reassigning items.

Examples

Moving a shipment through each step in the fulfillment flow requires making a series of PUT calls to the Shipment API

with the shipment number and the task being completed by the current step. These are generally very simple calls, with the appropriate task identified in the endpoint and only a basic request body to confirm the completion of that step.

Before you begin, refer back to Querying Order Shipments to learn how to retrieve shipments and their IDs for fulfillment.

Workflow Calls

Fulfillment steps are generally completed by calling workflow tasks from the Shipment API. For instance, accepting a shipment will consist of making a call with "Accept Shipment" as the specific task and "completed" as the action to progress through that step. If necessary, similar requests can be made to revert (undo) and skip the current step.

Though these exact requests are specified in each step of the following fulfillment examples, these base calls are outlined in the API documentation specs:

- Complete Workflow Task
- Get Workflow Tasks
- Get Workflow Task Counts
- Revert Workflow Task
- Skip Workflow Task

STH Fulfillment

Fulfilling a standard Ship to Home shipment will follow the below actions:

1. Accept Shipment
2. Validate Stock
3. Print Packing Slip
4. Add Tracking Information
5. Prepare for Shipment
6. Mark As Fulfilled

In all of these events, the API will return a 200 OK response code if the validation request was successful as well as the shipment data (such as is returned from a GET call to the Shipments API).

Accept Shipment

The first step is for the assigned fulfillment location to accept the shipment. Making the below call will mark it as accepted and allow the fulfiller to begin preparing the items and package.

```
ttps://t./api/commerce/shipments/{shipmentNumber}/tasks/Accept Shipment/completed
```

```
{
  "taskBody": {
    "shipmentAccepted": true
  }
}
```

Validate Stock

After acceptance, the fulfiller needs to validate that they have all of the shipment items in stock. There are three possible scenarios for this step – the fulfiller has enough inventory available for the full shipment, the fulfiller only has partial inventory available (or some of the items cannot yet be fulfilled for some reason), or the fulfiller has no inventory in stock and must reject the shipment. The first two scenarios both use “completed” in the endpoint, but the partial stock scenario will use a different request body that identifies the unavailable items.

If the fulfiller has all inventory in stock, then the simplest version of the call will be made with the completed endpoint:

```
https://t./api/commerce/shipments/{shipmentNumber}/tasks/Validate Items In Stock/completed
```

```
{
  "taskBody": {
    "stockLevel": "IN_STOCK"
  }
}
```

If not all items are yet available for fulfillment, the stock level must indicate “partial” as well as whether a transfer should be created to provide the missing items are not.

```
https://t./api/commerce/shipments/{shipmentNumber}/tasks/Validate Items In Stock/completed
```

```
{
  "taskBody": {
    "stockLevel": "PARTIAL_STOCK",
    "createTransfer": true
  },
  "handleOption": {
    "blockAssignment": false,
    "items": [
      {
        "lineId": 1,
        "blockAssignment": false,
        "quantity": 1,
        "reason": {
          "reasonCode": "NoInventory",
          "moreInfo": ""
        }
      }
    ]
  }
}
```

If the fulfiller does not have any items in stock or must reject the shipment for some other reason, the call will be made to an endpoint with “rejected” instead of “completed” in the URL. The request will provide a reason for the rejection and indicate whether future assignment to this location should be blocked or not.

`https://t./api/commerce/shipments/{shipmentNumber}/rejected`

```
{
  "rejectedReason": {
    "reasonCode": "Store is closing."
  },
  "blockAssignment": true
}
```

Print Packing Slip

If at least some items are in stock and the fulfiller is continuing with the fulfillment process, the next step is to print the packing slip that will be included in the shipment package.

`https://t./api/commerce/shipments/{shipmentNumber}/tasks/Print Packing Slip/completed`

Unlike the other steps, this call contains only an empty request body.

```
{
}
```

Add Tracking Information

When a shipment has been prepared for the carrier and a tracking number has been generated, it should be added to the shipment data. This is done at the package level underneath the shipment as an an Edit Package call. To retrieve the ID of the package, you can once again use a GET call to pull shipment information or reference the packages from the shipment data in the response of one of the previous calls.

`https://t./api/commerce/shipments/{shipmentNumber}/packages`

You can add query parameters to the URL to update only the trackingNumbers field of the package data, rather than the entire package payload.

`https://t./api/commerce/shipments/{shipmentNumber}/packages/{packageId}?updateFields=trackingNumbers`

Then, the request body only needs to include the tracking number (though a `shippingMethodCode` and `shippingMethodName` can be specified if desired).

```
{
  "trackingNumbers": [
    "1Z999999999999"
  ]
}
```

Prepare for Shipment

Marking the Prepare for Shipment step as completed will cause the shipment to be marked Fulfilled.

`https://t./api/commerce/shipments/{shipmentNumber}/tasks/Prepare for Shipment/completed`

The request body includes two booleans that confirm whether the shipment was successfully accepted by the carrier

and can be considered fulfilled (in which case, both booleans will be false).

```
{
  "taskBody": {
    "back": false,
    "canceled": false
  }
}
```

Mark As Fulfilled

If desired, the shipment can be manually marked as fulfilled by making a call to the Shipment API, though not as a task action.

`https://t{tenantId}.{host}/api/commerce/shipments/{shipmentNumber}/fulfilled`

There is no request body required for this call, only the shipment number in the endpoint.

BOPIS Fulfillment

Fulfilling a standard Buy Online Pickup In Store (BOPIS, or Pickup) shipment will follow the below actions:

1. Accept Shipment
2. Print Pick List
3. Validate Stock
4. Provide to Customer
5. Mark As Fulfilled

In all of these events, the API will return a 200 OK response code if the validation request was successful as well as the shipment data (such as is returned from a GET call to the Shipments API).

Accept Shipment

The first step is for the assigned fulfillment location to accept the shipment. Making the below call will mark it as accepted and allow the fulfiller to begin preparing the items and package.

`https://t./api/commerce/shipments/{shipmentNumber}/tasks/Accept Shipment/completed`

```
{
  "taskBody": {
    "shipmentAccepted": true
  }
}
```

Print Pick List

Once accepted, the pick list must be printed so that the store associates can collect the shipment items.

`https://t./api/commerce/shipments/{shipmentNumber}/tasks/Print Pick List/completed`

Unlike the other steps, this call contains only an empty request body.

```
{
}
```

Validate Stock

After picking, the fulfiller must indicate whether they had all of the shipment items in stock or not. Both cases use the same endpoint, but if some items were not available then they must be identified in the request body. A transfer shipment can be created so that those items can be shipped to the pickup location by another fulfiller.

Like BOPIS, the request body varies depending on whether or not the full inventory is available. If the location has the full inventory available, then the simplest version of the call is made to the completed endpoint:

`https://t./api/commerce/shipments/{shipmentNumber}/tasks/Validate Items In Stock/completed`

```
{
  "taskBody": {
    "stockLevel": "IN_STOCK"
  }
}
```

If not all items are yet available for fulfillment, the stock level must indicate “partial” as well as whether a transfer should be created to provide the missing items are not.

`https://t./api/commerce/shipments/{shipmentNumber}/tasks/Validate Items In Stock/completed`

```
{
  "taskBody": {
    "stockLevel": "PARTIAL_STOCK",
    "createTransfer": true
  },
  "handleOption": {
    "blockAssignment": false,
    "items": [
      {
        "lineId": 1,
        "blockAssignment": false,
        "quantity": 1,
        "reason": {
          "reasonCode": "NoInventory",
          "moreInfo": ""
        }
      }
    ]
  }
}
```

Provide to Customer

Marking the Provide to Customer step as completed should automatically cause the shipment to be marked Fulfilled.

`https://t./api/commerce/shipments/{shipmentNumber}/tasks/Customer Pickup/completed`

If the customer did not accept their items for some reason, then the items can be returned to the store inventory but the shipment will still be marked as complete and closed.

```
{
  "taskBody": {
    "customerAccepted": true
  }
}
```

Mark As Fulfilled

If desired, the shipment can be manually marked as fulfilled by making a call to the Shipment API, though not as a task action.

`https://t{tenantId}.{host}/api/commerce/shipments/{shipmentNumber}/fulfilled`

There is no request body required for this call, only the shipment number in the endpoint.

Transfer Fulfillment

Fulfilling a standard Transfer shipment, in which a second location ships missing inventory to the pickup location, will follow the below actions:

1. Validate Stock
2. Print Packing Slip
3. Add Tracking Information
4. Prepare for Shipment
5. Validate Incoming Transfer

In all of these events, the API will return a 200 OK response code if the validation request was successful as well as the shipment data (such as is returned from a GET call to the Shipments API).

Validate Stock

After a transfer shipment is assigned to a fulfillment location (so that they can send items to a pickup location that did not have them in stock), the fulfiller needs to validate whether they have all of the items available or not. Both cases use the same endpoint, but if some items were not available then they must be identified in the request body. Then, another new transfer shipment can be created to fulfill the missing items.

If the location has all of the requested items available and does not need to generate another transfer, then the simplest version of the call is made to the completed endpoint:

`https://t./api/commerce/shipments/{shipmentNumber}/tasks/Validate Items In Stock/completed`

```
{
  "taskBody": {
    "stockLevel": "IN_STOCK"
  }
```

If not all items are yet available for fulfillment, the stock level must indicate “partial” as well as whether a transfer should be created to provide the missing items are not.

`https://t./api/commerce/shipments/{shipmentNumber}/tasks/Validate Items In Stock/completed`

```
{
  "taskBody": {
     "stockLevel": "PARTIAL_STOCK",
     "createTransfer": true
  },
  "handleOption": {
     "blockAssignment": false,
     "items": [
        {
           "lineId": 1,
           "blockAssignment": false,
           "quantity": 1,
           "reason": {
              "reasonCode": "NeedToChangePaymentMethod",
              "moreInfo": ""
           }
        }
     ]
  }
}
```

Print Packing Slip

If at least some items are in stock, the next step is to print the packing slip that will be included in the shipment package.

`https://t./api/commerce/shipments/{shipmentNumber}/tasks/Print Packing Slip/completed`

Unlike the other steps, this call contains only an empty request body.

```
{
}
```

Add Tracking Information

When a shipment has been prepared for the carrier and a tracking number has been generated, it should be added to the shipment data. This is done at the package level underneath the shipment as an an Edit Package call. To retrieve the ID of the package, you can once again use a GET call to pull shipment information or reference the packages from the shipment data in the response of one of the previous calls.

`https://t./api/commerce/shipments/{shipmentNumber}/packages`

You can add query parameters to the URL to update only the trackingNumbers field of the package data, rather than the entire package payload.

`https://t./api/commerce/shipments/{shipmentNumber}/packages/{packageId}?updateFields=trackingNumbers`

Then, the request body only needs to include the tracking number (though a `shippingMethodCode` and `shippingMethodName` can be specified if desired).

```
{
  "trackingNumbers": [
    "1Z999999999999"
  ]
}
```

Prepare For Shipment

Marking the Prepare for Shipment step as completed should cause the shipment to automatically be marked Fulfilled.

`https://t./api/commerce/shipments/{shipmentNumber}/tasks/Prepare for Shipment/completed`

The request body includes two booleans that confirm whether the shipment was successfully accepted by the carrier and can be considered fulfilled (in which case, both booleans will be false).

```
{
  "taskBody": {
    "back": false,
    "canceled": false
  }
}
```

Validate Incoming Transfer

Once the transfer shipment has been received at the pickup location, its stock must be validated to ensure that it includes the correct items and quantities that were requested. Similarly to the initial Validate Stock step, the request body can either indicate that all items were made available or it can indicate that some items are still missing and trigger a new transfer shipment. Both cases use the same parameter schema as the initial Validate Stock step, but have a different endpoint.

`https://t./api/commerce/shipments/{shipmentNumber}/tasks/Validate Incoming Transfer/completed`

For validation of the full inventory, the request body simply confirms the stock level:

```
{
  "taskBody": {
    "stockLevel": "IN_STOCK"
  }
}
```

If not all items are yet available for fulfillment then the stock level must indicate “partial.”

```
{
  "taskBody": {
    "stockLevel": "PARTIAL_STOCK",
    "createTransfer": true
  },
  "handleOption": {
    "blockAssignment": false,
    "items": [
      {
        "lineId": 2,
        "blockAssignment": false,
        "quantity": 1,
        "reason": {
          "reasonCode": "NoInventory",
          "moreInfo": ""
        }
      }
    ]
  }
}
```

Other Fulfillment Calls

There are additional calls that can be made to the Shipments API to perform different events in the fulfillment process. In all cases, the full shipment data will be returned in the response just like it is for a Get Shipment call.

- Backorder Shipment: POST /commerce/shipments/{shipmentNumber}/backordered
- Cancel Shipment: PUT /commerce/shipments/{shipmentNumber}/canceled
- Reassign Shipment: PUT /commerce/shipments/{shipmentNumber}/reassigned
- Create Transfer Shipment: PUT /commerce/shipments/{shipmentNumber}/transferred
- Receive Transfer Shipment: PUT /commerce/shipments/{shipmentNumber}/received
- Reject Shipment: PUT /commerce/shipments/{shipmentNumber}/rejected

The following Shipment APIs can be used in conjunction with Future Inventory:

- Create Future Shipments for Items: POST /commerce/shipments/{shipmentNumber}/futureItems
- Future Shipment to Ready (converts a shipment from Future status to Ready): PUT /commerce/shipments/{shipmentNumber}/futureToReady
- Update Future Shipment Date: PUT /commerce/shipments/-shipmentNumber-/futureUpdateDate

Shipment Note Calls

Shipment notes are unique and separate from order notes. They are not copied onto any child shipments or order-level data, and so will only be displayed for the shipment they were originally added to. A shipment note cannot be created or edited in the Fulfiller UI, only via the Shipments API through the following endpoints.

- Create Shipment Note: POST /commerce/shipments/{shipmentId}/notes
- Update Shipment Note: PUT /commerce/shipments/{shipmentId}/notes/{noteId}

- Delete Shipment Note: DELETE /commerce/shipments/{shipmentId}/notes/{noteId}

The request to create or update a note should define the note text, the username, and role of the user creating the note as shown below. The timestamp will be automatically recorded.

```
{
  “noteText”: “Sample Note Text”,
  “role”: “Superadmin”,
  “username”: “Kibo 123”
}
```

This data will be saved and returned in a `shipmentNotes` object from a GET Shipment call. The API documentation is not currently updated with these endpoints or the `shipmentNotes` object due to technical issues (as of May 2022).

Item Fulfillment Calls

Some of the same calls made at the shipment level can be applied to items within the shipments, such as in a case where part of the inventory was rejected or picked up by the customer. In these cases, the request payloads should only contain the items being backordered, canceled, etc. from the location. There is no need to include items that the particular action is not being performed on.

- Backorder Items: POST /commerce/shipments/{shipmentNumber}/backorderedItems
- Cancel Items: PUT /commerce/shipments/{shipmentNumber}/canceledItems
- Pick Up Items: POST /commerce/shipments/{shipmentNumber}/pickedUpItems
- Reassign Items: PUT /commerce/shipments/{shipmentNumber}/reassignedItems
- Reject Items: PUT /commerce/shipments/{shipmentNumber}/rejectedItems
- Transfer Items: PUT /commerce/shipments/{shipmentNumber}/transferredItems

Note that items can’t be rejected once they have been moved to a fulfilled state.